Cracking Hashed Passwords

Let‘s start by cracking some hashed passwords from security breaches.

First, create a file `hashes.txt` containing hashes for common passwords like "linkedin" and "password":

```
$ echo "b1cd58cb38eb273ab24b50c47de81cf4" >> hashes.txt
$ echo "5baa61e4c9b93f3f0682250b6cf8331b7ee68fd8" >> hashes.txt
```

These are MD5 hashes, so use hash mode 0:

```
$ hashcat -m 0 hashes.txt -a 0 wordlist.txt
```

Hashcat cracks "password" instantly! For "linkedin", make a bigger wordlist and add rules:

```
$ hashcat -m 0 hashes.txt -r rules/best64.rule merged_wordlist.txt
```

The rules mutate "linkedin" into "LinkedIn" and crack the hash.

Cracking Hashed WPA Handshakes

Cracking WPA2 Wi-Fi handshakes recovered from .cap files gives the passphrase needed to decrypt wireless traffic.

First, use cap2hccapx to convert the .cap into Hashcat‘s hccapx format.

Then run a mask attack brute-forcing 8 lower characters + 2 digits:

```
$ hashcat -m 2500 handshake.hccapx -a 3 ?l?l?l?l?l?l?l?l?d?d
```

Once cracked, we can decrypt the Wi-Fi traffic in Wireshark using the pre-shared key.

Unlocking Encrypted Containers

Many encrypted container formats like VeraCrypt use PBKDF2 and AES. We can crack VeraCrypt containers by attacking the keyfile.

Use VeraCrypt to export the encrypted keyfile. Then load it into Hashcat:

```
$ hashcat -m 15600 veracrypt_keyfile.hc -a 0 dictionary.txt
```

This tries cracking the keyfile with a dictionary attack. Once succeeded, we can mount the VeraCrypt container with the passphrase.

Auditing Password Security

Hashcat can also generate password hashes for auditing purposes.

Say you want to assess how your users‘ passwords fare against cracking. Generate hash dumps for common passwords:

```
$ echo "P@ssw0rd" | hashcat --stdout -m 1800
```

This returns the SHA-512 crypt hash, which can be saved and cracked offline to audit security.